**<<LAB 10: Dictionaries and Sets>>**

|  |  |  |  |
| --- | --- | --- | --- |
| Course Title | **Programming Fundamentals** | Course Code | **CC1021** |
| Lab Time | **2.5 hours** | Section | **##** |

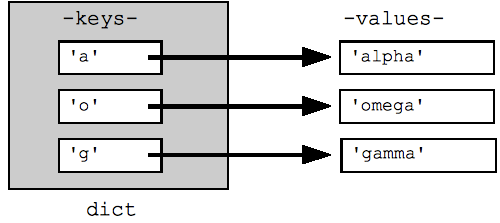
**Instructor’s Name and Contact Information:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | **XYZ** | Office No. | **##** |
| E-mail | **xyz@umt.edu.pk** | Office Tel. | **EXT** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Course Learning Outcomes (CLOs)** | **Domain** | **BT Level** | **Corresponding Program Learning Outcomes (PLOs)** |
| * 1. **Apply** basic programing concepts | C | 3 |  |
| * 1. **Design** and implement algorithms to solve real world problems. | C | 3 |  |

**DICTIONARIES**

Dictionaries are Python’s implementation of a data structure that is more generally known as an associative array. A dictionary consists of a collection of key-value pairs. Each key-value pair maps the key to its associated value.



**Defining a dictionary**

We can define a dictionary by enclosing a comma-separated list of key-value pairs in curly braces “{}”. A colon (:) separates each key from its associated value

|  |
| --- |
| d = {  <key>: <value>,  <key>: <value>,  .  .  <key>: <value>  } |

**Properties of Dictionary Keys**

Dictionary values have no restrictions. They can be any arbitrary Python object, either standard objects or user-defined objects. However, same is not true for the keys.

There are two important points to remember about dictionary keys:

1. More than one entry per key not allowed. Which means no duplicate key is allowed.
2. Keys must be immutable. Which means you can use strings, numbers or tuples as dictionary keys but something like ['key'] is not allowed.

**Example:**

|  |
| --- |
| Define and print a dictionary having keys (1,2,3) and values (“red”, “blue”, ”green”)  **Solution:**  **Output:** |

We can also construct a dictionary with the built-in dict() function. The argument to dict() should be a sequence of key-value pairs. A list of tuples works well for this

**Example:**

|  |
| --- |
| Defining a dictionary using dict() |

If the key values are simple strings, they can be specified as keyword arguments. So here is yet another way to define dictionary.

**Example:**

|  |
| --- |
|  |

**Accessing Dictionary Items**

The entries in the dictionary display in the order they were defined. But that is irrelevant when it comes to retrieving them. Dictionary elements are not accessed by numerical index

|  |
| --- |
|  |

A value is retrieved from a dictionary by specifying its corresponding key in square brackets "[]"

**Example:**

|  |
| --- |
|  |

If we refer to a key that is not in the dictionary, Python raises an exception

**Example:**

|  |
| --- |
|  |

**Adding items in dictionary**

Adding an item to the dictionary is done by using a new index key and assigning a value to it

**Example:**

|  |
| --- |
|  |

The update() method will update the dictionary with the items from a given argument. If the item does not exist, the item will be added.

The argument must be a dictionary, or an iterable object with key: value pairs.

**Example:**

|  |
| --- |
|  |

**Change Dictionary items**

You can change the value of a specific item by referring to its key name

The update() method will update the dictionary with the items from a given argument. If the item does not exist, the item will be added.

The argument must be a dictionary, or an iterable object with key: value pairs.

**Example:**

|  |
| --- |
|  |

**Remove Dictionary Items**

* The pop() method removes the item with the specified key name
* The popitem() method removes the last inserted item
* The del keyword removes the item with the specified key name

**Example:**

|  |
| --- |
|  |

* The del keyword can also delete the dictionary completely

**Example:**

|  |
| --- |
|  |

* The clear() method empties the dictionary

|  |
| --- |
|  |

**Loop through Dictionary**

When looping through a dictionary, the return value are the keys of the dictionary, but there are methods to return the values as well.

**Example:**

* Print all key names in the dictionary, one by one

|  |
| --- |
|  |

**Example:**

* Print all values in the dictionary, one by one

|  |
| --- |
|  |

**Example:**

* The values() method to return values of a dictionary

|  |
| --- |
|  |

**Example:**

* The keys() method to return values of a dictionary

|  |
| --- |
|  |

**Example:**

* The items() method to return both keys and values of a dictionary

|  |
| --- |
|  |

**Nested Dictionaries**

A dictionary can contain dictionaries; this is called nested dictionaries. Also it can contain any iterable objects as values.

**Example:**

|  |
| --- |
|  |

**Access Nested Dictionaries items**

In order to access the value of any key in the nested dictionary, we will use indexing [] syntax. Also, the syntax will remain the same to print the items one by one.

**Example:**

|  |
| --- |
|  |

**SETS**

Sets are used to store multiple items in a single variable. Set is one of 4 built-in data types in Python used to store collections of data, the other 3 are List, Tuple, and Dictionary, all with different qualities and usage.

A set is a collection which is unordered, unchangeable, and do not allow duplicate values.

|  |  |
| --- | --- |
| Unordered | It means that the items in a set do not have a defined order.  Set items can appear in a different order every time you use them, and cannot be referred to by index |
| Unchangeable | It means that we cannot change the items after the set has been created.  Once a set is created, you cannot change its items, but you can remove items and add new items. |
| do not allow duplicate values | It means sets cannot have two items with the same value. |

**Defining a set**

A set can be defined using curly brackets ( {} ) and by a built-in method named set, which will take an iterable object to create a set.

|  |
| --- |
|  |

**Do not allow duplicates**

sets cannot have two items with the same value

|  |
| --- |
|  |

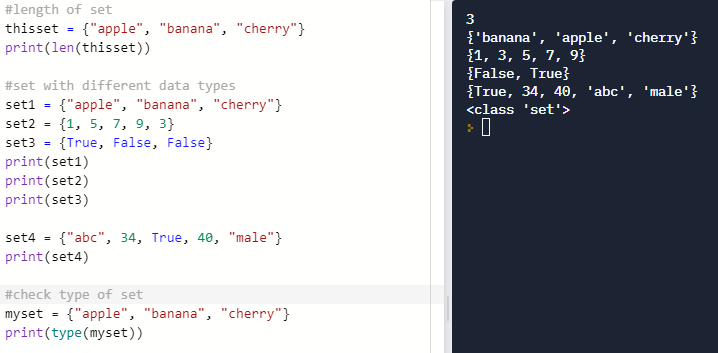
**Length of a Set**

To determine how many items a set has, use the len() method.

**Set Items’ Data Types**

Set items can be of any data type.

**Example:**



**Accessing Set Items**

You cannot access items in a set by referring to an index or a key. But you can loop through the set items using a for loop, or ask if a specified value is present in a set, by using the in keyword.

**Example:**

|  |
| --- |
|  |

**Membership of Items in Sets**

It means an item exists in set or not.

**Example:**

|  |
| --- |
|  |

**Add Items**

Once a set is created, you cannot change its items, but you can add new items. To add one item to a set use the add() method Also, To add items from another iterable object into the current set, use the update() method.

**Example:**

|  |
| --- |
|  |

**Remove Items**

To remove an item in a set, use the remove(), or the discard() method. We can also use the pop() method to remove an item, but this method will remove the last item. Remember that sets are unordered, so you will not know what item that gets removed. The return value of the pop() method is the removed item. The clear() method empties the set and the del keyword will delete the set completely

**Example:**

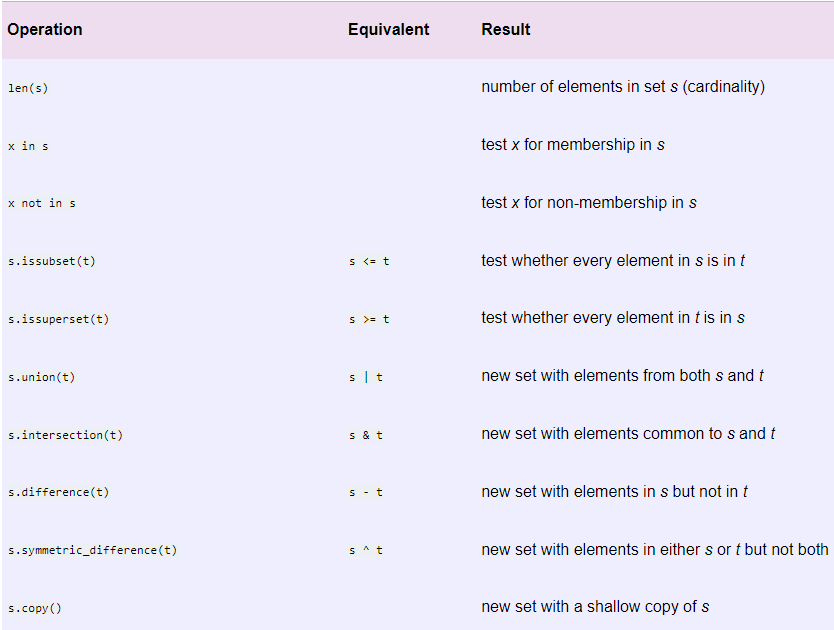
|  |
| --- |
|  |

**Example:**

|  |
| --- |
|  |

**Operations over set**

Operations over sets are same recalling the mathematical sets and set theories. Following snippet shows operations over set:



**LAB TASKS**

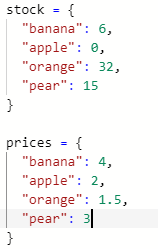
Task 1:

|  |
| --- |
|  |

Perform the Following for the dictionary given above.

* Display the keys using loop
* Display the values of dictionary by using the suitable method
* Add a key “LowestScore” having the value 10.
* Remove the key “TypeofMatchesPlayed”
* Add a new key “CategoryPlayed” having value as a dictionary for which keys and values will be taken from the user in lists named “category” and “MatchesPlayed”
* Display the updated dictionary as key values pairs.

|  |
| --- |
|  |

Task 2:

* Define two lists given in snippet
* Write python code to compute the bill for shopping list as Dictionary given by user.
* Shopping list will have available items as keys and their quantity as values.
* While you loop through each item of shopping list, only add the price of the item to total if the item's stock count is greater than zero.
* If the item is in stock and after you add the price to the total, subtract one from the item's stock count.
* Display the total for the shopping list

|  |
| --- |
|  |

Task 3:

We have following information on Employees and their Salary (Salary is in lakhs),

|  |  |
| --- | --- |
| Employee | Salary |
| John | 14 |
| Smith | 13 |
| Alice | 32 |
| Daneil | 21 |

Using above create a dictionary of Employees and their Salary

Write a program that asks user for three type of inputs,

1. **print**: if user enter print then it should print all Employees with their Salary in this format,

|  |
| --- |
| Employee Salary  John 14  Smith 13  Alice 32  Daneil 21 |

1. **add:** if user input adds then it should further ask for an Employee name to add. If Employee already exists in our dataset then it should print that it exists and do nothing. If it doesn't then it asks for Salary and add that new Employee/Salary in our dictionary and print it
2. **remove:** when user inputs remove it should ask for an Employee to remove. If an Employee exists in our dictionary then remove it and print a new dictionary using format shown above in (a). Else print that Employee doesn't exist!
3. **query:** on this again ask the user for which Employee he or she wants to query. When a user inputs that Employee it will print the Salary of that Employee.
4. **Exit:** exit the program

|  |
| --- |
|  |

Task 4:

Using loops perform the following operations over sets:

1. Union of two sets
2. Intersection of two sets
3. Difference of two sets
4. Check if one set is subset of another or not

|  |
| --- |
|  |

Task 5:

Write a python program to find the maximum and minimum value in a set of integers using loops

|  |
| --- |
|  |

Task 6:

Write a python program to accept the strings which contains all vowels.

|  |
| --- |
| Input : geeksforgeeks  Output : Not Accepted  All vowels except 'o' are not present  Input : ABeeIghiObhkUul  Output : Accepted  All vowels are present |

|  |
| --- |
|  |